Assignment 2: Text Classification

Natasa Farmaki - DS3517018

Dimitris Georgiou - DS3517004

Stratos Gounidellis - DS3517005

Course: Text Engineering & Analytics

Professor: Ion Androutsopoulos

Athens, March 2018

INTRODUCTION

The aim of this assignment is to classify a set of email as legitimate or spam. It is a fact that unsolicited commercial or bulk e-mail also known as spam constitute a big trouble over the internet. Spam are considered waste of time, storage space and communication bandwidth. Thus, spam filtering is way to ameliorate the situation. More specifically, spam filtering and detection is the process of detecting spam emails and prevents those messages from getting to a user's inbox.

In our approach, we use three different techniques to classify the emails in the two categories, i.e. legitimate and spam emails. To be more specific we employed Naïve Bayes, Logistic Regression and k-nearest neighbor algorithms. In order to evaluate their effectiveness, we utilized different metrics, such as accuracy, precision, recall and F1-score because the dataset is imbalanced as the spam emails are almost the half of the legitimate emails. In addition, we compared the three approaches with a simplistic baseline approach and we generated learning curves and precision-recall curves.

The python code is available in the following link: https://thinkingtea.github.io/TEArepo/

CORPUS COLLECTION

The Enron Corpus is a large database of over 600,000 emails generated by 158 employees of the Enron Corporation and acquired by the Federal Energy Regulatory Commission during its investigation after the company's collapse. This dataset is quite interesting as it is one of the few large email corpus available in the public domain.

The Enron data was originally collected at Enron Corporation headquarters in Houston during two weeks in May 2002 by Joe Bartling, a litigation support and data analysis contractor working for Aspen Systems, whom the Federal Energy Regulatory Commission (FERC) had hired to preserve and collect the vast amounts of data in the wake of the Enron Bankruptcy in December 2001.

In the current subset, we used a subset of the above dataset with 5172 pre-processed emails (3672 legitimate emails and 1500 spam emails). That subset is available here.

CORPUS PREPROCESSING

Firstly, we split the emails in a training and test set, with the latter equal to the thirty percent (30%) of the initial dataset (3620 training data and 1552 test data). Additionally, the label is encoded with 0 and 1, with 0 standing for "legitimate" and 1 standing for "spam". In addition, we convert the collection of the emails to a sparse matrix of TF-IDF features. We use unigrams, bigrams and trigrams. When building the vocabulary, we chose to ignore terms that have a document frequency strictly lower than a given threshold, which in our case is set to ten (10). This value is also called cut-off in the literature. In that way, we remove the insignificant words, which appear in the sentences and do not have any meaning or indications about the content.

Apart from the TF-IDF features, we utilize also the length of each email as a feature. Then we normalize the features to be between -1 and 1. This is a significant pre-processing step especially for the SVD method following later.

In order to determine which features are of paramount importance for the effectiveness of the classifiers we use both Random Forest Classifier and select k best features of sklearn library. From each of these methods we choose the top 20 features and use the union of those features without preserving duplicates.
Then, we apply SVD on the TF-IDF matrix. More specifically, in linear algebra, the singular-value decomposition (SVD) is a factorization of a real or complex matrix. It is the generalization of the

eigendecomposition of a positive semidefinite normal matrix (for example, a symmetric matrix with positive eigenvalues) to any m X n matrix via an extension of the polar decomposition. It has many useful applications in signal processing and statistics. To implement SVD we need to determine the number of components that we want to keep. To find the optimal number of these components we use the variance as criterion. So we create a function that takes as input the variance of the initial data that we want svd-components to explain and returns the number of features to choose. We keep enough components to explain at least 80% of the variance (rule of thumb).

Finally, we combine the SVD components with the features selected with the two methods mentioned above. The final number of features used to predict the label of the email is 715.

CLASSIFICATION OF EMAIL MESSAGES

In order to construct an efficient model, we implement multiple classifiers and utilize appropriate metrics in order to determine the most effective one.

Baseline approach
As stated above, the dataset is imbalanced, which means that the legitimate messages are far more than the spam messages. Therefore, it is useful to construct a baseline classifier that would just classify all emails as legitimate. In other words, in the baseline algorithm, legitimate messages are (correctly) never blocked and spam messages (mistakenly) always pass.

Logistic Regression
Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable. The procedure is quite similar to multiple linear regression, with the exception that the response variable is binomial. The result is the impact of each variable on the odds ratio of the observed event of interest.

In other words, logistic regression constitutes a tool for building a model in situations where there is a two-level categorical response variable, in contrast to a numerical response variable, where multiple linear regression would be more appropriate. Like multiple regression, logistic regression is a type of generalized linear model (GLM) with the difference being the categorical response variable. The outcome of a GLM is usually denoted by Y_i, where i stands for observation number i. In our case, Y_i denotes whether an email is spam or legitimate, ($Y_i = 1$) for spam, and ($Y_i = 0$) for legitimate.

Naive Bayesian classification
Naive Bayes as its name indicates is based on a simple, ”naive” implementation of Bayes rule. Bayes theorem provides a way of calculating the posterior probability of a class c (spam, legitimate) given an email composed of words x. Nevertheless, in the case of Naive Bayes classifier it is assumed that the effect of the value of a predictor (*x*) on a given class (*c*) is independent of the values of other predictors.

$$P(c \mid x) = \frac{P(x \mid c) * P(c)}{P(x)}$$

This assumption is of course an oversimplification, which is hardly ever true in real problems. However, it seems that Naive Bayes classifier often does surprisingly well outperforming more sophisticated classification methods while providing advantages of limited computational needs and faster execution of the algorithm.

k-Nearest Neighbors
The k-Nearest Neighbors algorithm, also known as kNN, is a non-parametric method used for pattern classification. It is an instance-based learning algorithm. The function is simply approximated locally.

The neighbors for the data points are taken as a set of objects whose values are known. This is the training phase of the algorithm. In the classification phase of the algorithm, for each point in the testing set, we consider k neighbors where the analyst / data scientist predefines k.

MEASURES TO EVALUATE CLASSIFICATION PERFORMANCE

In order to evaluate the results of the multiple classifiers and therefore compare their performance we need to implement appropriate measures. As our problem is imbalanced, accuracy is not a good measure because a simple majority classifier that always predicts that an email is legitimate will get an accuracy of 80%. For this reason apart from accuracy for each classifier, we will calculate precision, recall, F1-score metrics and Total Cost Ratio.

Accuracy is the number of correct predictions made divided by the total number of predictions made, multiplied by one hundred (100) to turn it into a percentage. In order to calculate the total cost ratio mentioned below we also calculate **error rate**, which is one minus Accuracy.

Precision is the number of True Positives divided by the number of True Positives and False Positives. Precision can be considered a measure of a classifier's exactness. A low precision can also indicate a large number of False Positives.

Recall is the number of True Positives divided by the number of True Positives and the number of False Negatives. In other way, it is the number of positive predictions divided by the number of positive class values in the test data. It is also known as Sensitivity or the True Positive Rate. Recall is a measure of a classifiers completeness. A low recall indicates many False Negatives.

A system with high recall but low precision returns many results, but most of its predicted labels are incorrect when compared to the training labels. A system with high precision but low recall is just the opposite, returning very few results, but most of its predicted labels are correct when compared to the training labels. An ideal system with high precision and high recall will return many results, with all results labeled correctly.

F1-score is the 2 * ((precision * recall) / (precision + recall)). In other words, the F1-score conveys the balance between the precision and the recall.

Total Cost Ratio (TCR) is a metric that allows the performance of a classifier-filter to be compared to that of the baseline. It is defined as the weighted error of the baseline (1- weighted accuracy) divided by the weighted error of the classifier we wish to compare with the baseline.

$$Werr^b = \frac{N_{spam}}{\lambda * N_{legit} + N_{spam}}$$

$$Werr = \frac{\lambda * n_{legit \to spam} + n_{spam \to legit}}{\lambda * N_{legit} + N_{spam}}$$

$$TCR = \frac{Werr^b}{Werr}$$

ROC curve

In statistics, a receiver operating characteristic curve, i.e. ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability

of detection. The false-positive rate is also known as the fall-out or probability of false alarm . The ROC curve is thus the sensitivity as a function of fall-out.

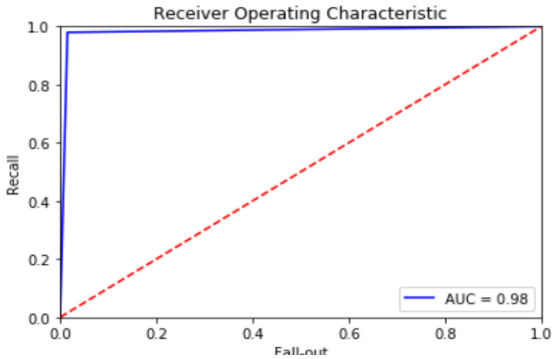
Confusion Matrix
A way to present the prediction results of a classifier is to use a confusion matrix. In this matrix, the number of correct and incorrect predictions are summarized with count values and are broken down by class. In fact, the confusion matrix shows the ways in which the classification model is confused when it makes predictions. It offers valuable insight not only into the errors being made by the classifier but more importantly the types of errors that are being made. The matrix is extracted by a truth table. For a binary classification problem, the truth table has two (2) rows and two (2) columns. Across the top is the observed class labels and down the side are the predicted class labels. Each cell contains the number of predictions made by the classifier that fall into that cell.

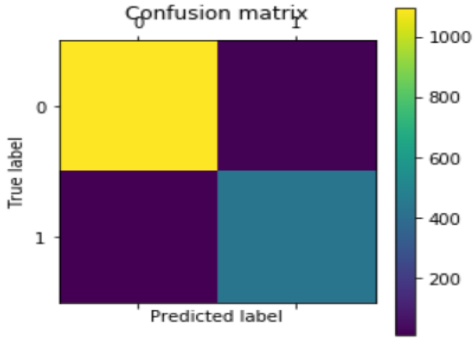
Learning Curve
A learning curve is the representation in graph form of the rate of learning something over time or repeated experiences. In a text classification problem, a learning curve shows whether collecting more training instances will improve the performance of the classifier both in training and test set. It is important to note that learning curves are not useful for model assessment.

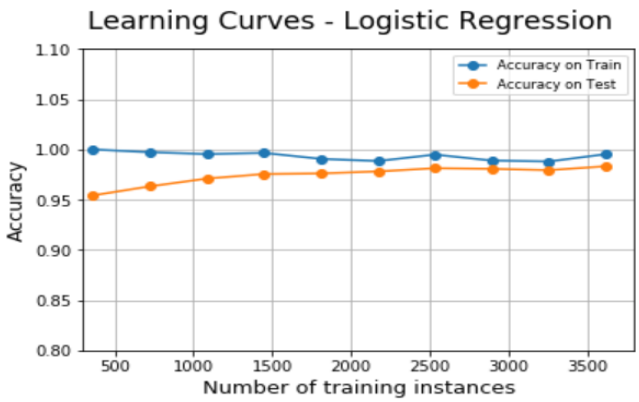
Precision -Recall curve
Precision-Recall curve is a useful measure of success of prediction when the classes are very imbalanced. The precision-recall curve shows the tradeoff between precision and recall for different threshold. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall).

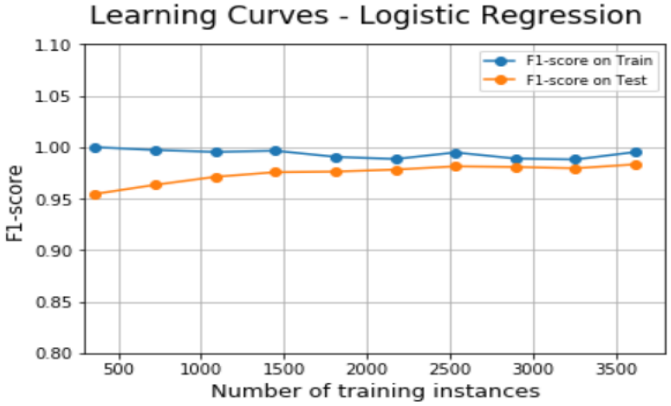
EXPERIMENTAL RESULTS

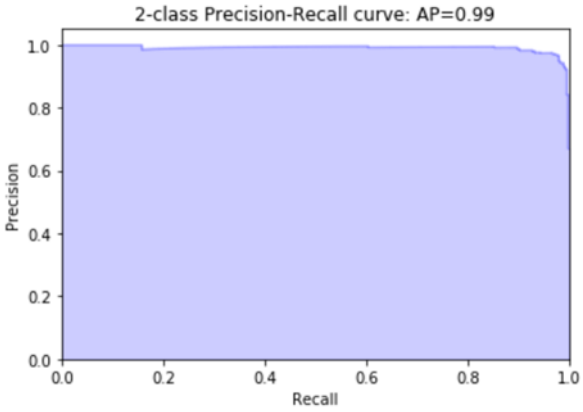
Logistic Regression









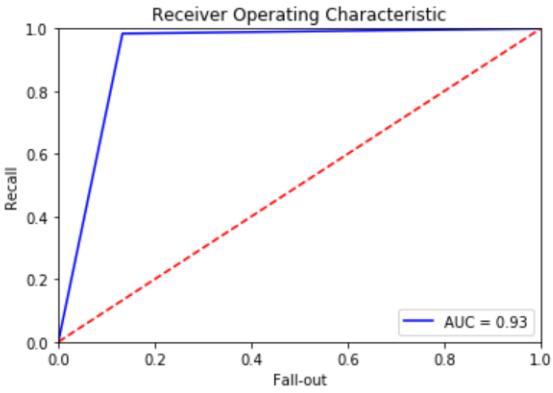


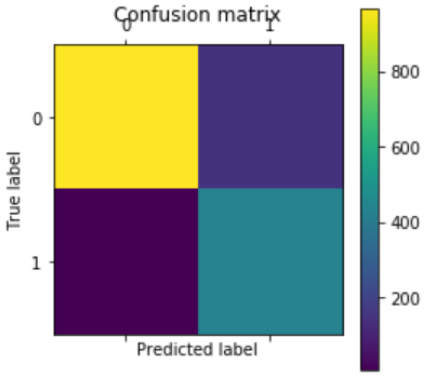

```
Accuracy 0.978172580063 [ 0.98344828  0.9820442   0.97790055  0.97928177  0.96818811]
Precision 0.951441872455 [ 0.97183099  0.95022624  0.95794393  0.94570136  0.93150685]
Recall 0.975498272655 [ 0.97183099  0.99056604  0.96698113  0.98584906  0.96226415]
F1 0.963248580861 [ 0.97183099  0.96997691  0.96244131  0.96535797  0.94663573]
```

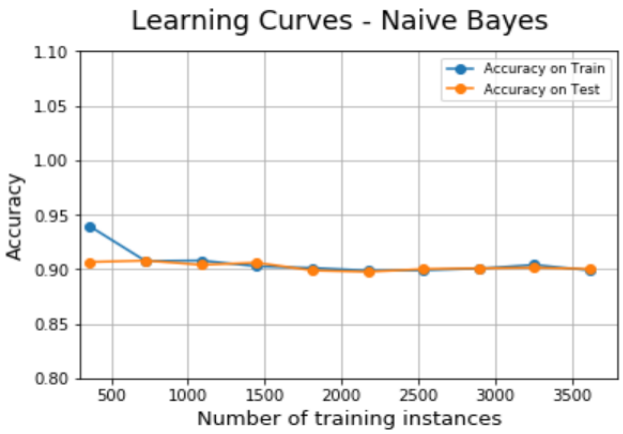
5-fold metrics (Accuracy - Precision - Recall - F1-score)

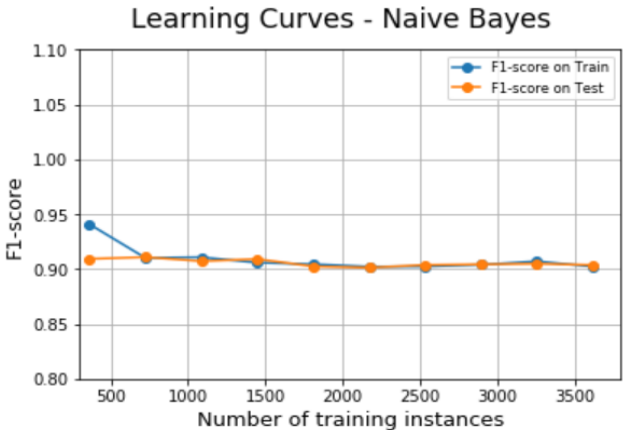
Total cost ratio - Logistic Regression vs. baseline classifier: 10.2093023256

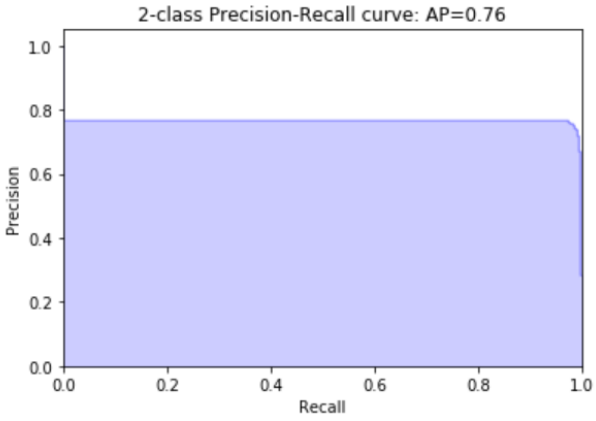
Naive Bayesian classification









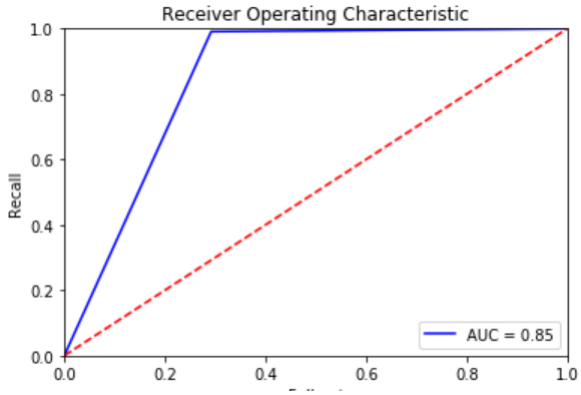


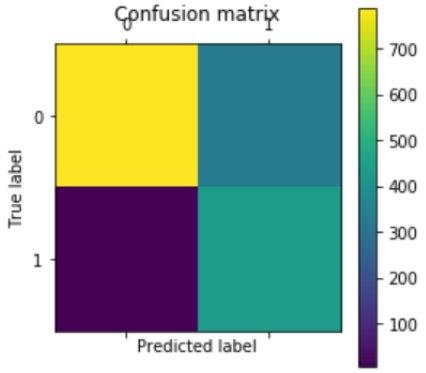

```
Accuracy 0.897235580669 [ 0.89793103  0.89226519  0.89364641  0.91160221  0.89073306]
Precision 0.748190167133 [ 0.75090253  0.73591549  0.74907749  0.77007299  0.73498233]
Recall 0.97926742847 [ 0.97652582  0.98584906  0.95754717  0.99528302  0.98113208]
F1 0.848203607014 [ 0.84897959  0.84274194  0.84057971  0.86831276  0.84040404]
```

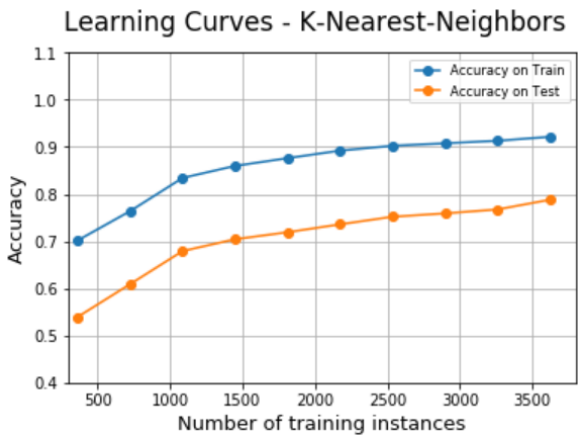
5-fold metrics (Accuracy - Precision - Recall - F1-score)

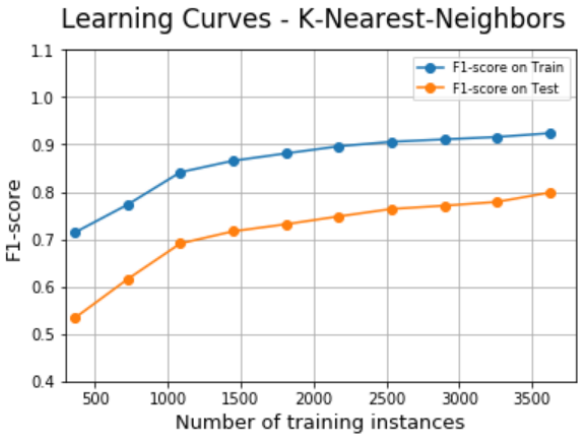
Total cost ratio - Naive Bayes vs. baseline classifier: 1.44884488449

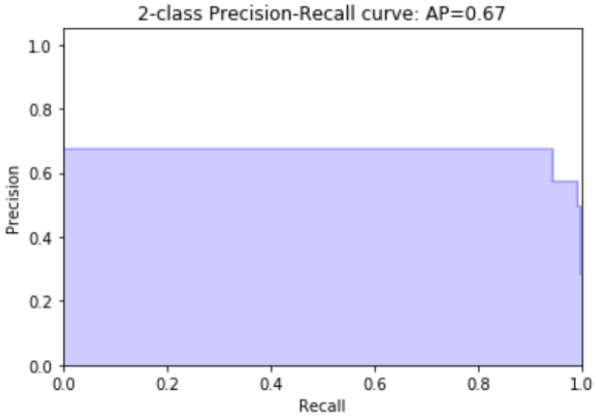
k-Nearest Neighbors












```
Accuracy 0.816572924514 [ 0.8262069   0.79005525  0.82458564  0.8218232   0.82019364]
Precision 0.617076932675 [ 0.62908012  0.58426966  0.62686567  0.6231454   0.62202381]
Recall 0.988683674373 [ 0.99530516  0.98113208  0.99056604  0.99056604  0.98584906]
F1 0.759785799879 [ 0.77090909  0.73239437  0.7678245   0.76502732  0.76277372]
```

5-fold metrics (Accuracy - Precision - Recall - F1-score)

Total cost ratio - knn vs. baseline classifier: 0.67125382263

CONCLUSIONS

As it is obvious from the above analysis, the logistic regression analysis seems to outperform. The Total Cost Ratio is significantly greater than the Total Cost Ratio of the Naïve Bayesian classification and of the k-Nearest Neighbors approach, which seems to be even worse than the baseline approach. Additionally, the logistic regression classifier has better precision-recall curves in comparison to the other two techniques and thus it seems to be the best approach in the given dataset. Of course, more emails should be taken into account for more accurate comparison among the three techniques.